Методические рекомендации по разбору предложенных олимпиадных задач

Муниципального этапа Всероссийской олимпиады школьников по информатике в 2023-2024 учебном году

7- 8 класс

A. Поиск подстроки

В первом вопросе нужно посчитать количество появлений строки из 51 букв «Я» внутри строки из 100 букв «Я». Это можно сделать 50 способами: до выбранной строки может идти от 0 до 49 букв «Я».

Во втором вопросе строка «ОХО» входит два раза в строку «ХОХОХО», и ещё одно вхождение образуется “на границе” между двумя вхождениями. Итого 299 вхождений.

В третьем вопросе строка «МА» входит 2 раза в строку «МАТЕМАТИКА», если её повторить 100 раз, то число повторений будет равно 200.

В четвертом вопросе строка «АК» входит один раз в строку «КАРАКАТИЦА» (всего 100 раз), и ещё один раз входит на границу между двумя строками «КАРАКАТИЦАКАРАКАТИЦА», что даёт ещё 99 вхождений. Итого 199 вхождений.

В пятом вопросе, если строку «МУРМУР» повторить 100 раз, то получится 200 повторений слога «МУР». Нам нужно посчитать число вхождений «УРМУРМУР», для этого нужно три слога идущих подряд. Это можно сделать 198-ю способами.

Правильный ответ:
50
299
200
199
198

B. День рождения

Обозначим число первого вида конфет за x, а второго – за y. Можно заметить, что сумма $0 < (x + y) < 2N$ и делится на N, значит, она равна N. Это значит, что для каждого конкретного значения x мы можем точно сказать, что $y = N - x$. Тогда можно перебрать все x от 0 до d и проверить, что пара $(x, N - x)$ является корректной.
Должны выполняться неравенства:

$$0 \le x \le d,$$
$$0 \le N - x \le t.$$

Тогда

$$0 \le x \le d,$$
$$x \ge N - t.$$

Тогда ответ на задачу – формула max(0, d - (N - t) + 1).

Пример решения.

```
n = int(input())
d = int(input())
t = int(input())
print(max(0, d - (n - t) + 1))
```

C. Офис

Решение

```
a = int(input())
k = 0
while a>=2:
    k += 1
    a -= a//2
print(k)
```

D.Образовательный канал

Заведём массив present из N элементов (при нумерации элементов массива с нуля удобней завести массив из N + 1 элемента), отмечая в нём, просмотрен ли этот выпуск или нет. Первоначально все элементы этого массива будут иметь значение «Ложь», а после считывания числа k присвоим k-му элементу массива значение «Истина». Потом пройдём по всему массиву и выведем индекс такого элемента, значение которого осталось «Ложь».

Пример такого решения.

```
n = int(input())
present = [False] * (n + 1)
for i in range(n - 1):
    k = int(input())
    present[k] = True
for i in range(1, n + 1):
    if not present[i]:
        print(i)
```

Но у этой задачи есть и красивое решение, не использующее массивов. Для этого заметим, что если рассмотреть сумму чисел 1 + 2 + ... + n и вычесть данные числа, то получится как раз недостающее число.

Пример такого решения (к переменной s добавляются числа от 1 до n, и вычитаются данные числа):

```
n = int(input())
s = 0
for i in range(1, n):
    s += i
    s -= int(input())
s += n
print(s)
```

E. Кинотеатр

Эффективное решение включает в себя формулы для вычисления номера ряда и номера места. Для удобства номер билета будем отсчитывать с нуля, для этого вычтем из N единицу. Известно, что каждые x+y рядов содержат x*a+y*b мест. Посмотрим, сколько раз укладывается x + y в N. Разделим номер билета N на (x*a + y*b) с остатком. Обозначим частное за d, остаток за r. Рассмотрим остаток r, если остаток r от 0 до x*a – 1, то к номеру ряда добавится add = r / a и ряд будет равен N / (x*a + y*b) * (x+y) + r/a + 1, а место будет равно r % a + 1, иначе добавится add = x + (r – x*a)/b и ряд будет равен N / (x*a + y*b) * (x+y) + add + 1, а место будет равно r % b + 1.

```
#include <iostream>
using namespace std;
int main(){
 long long x, a, y, b, N;
 cin >> x >> a >> y >> b >> N;
 long long row = 0;
 long long seat = 0;

 N -= 1;
 row = N / (x*a + y*b) * (x+y);
 N = N % (x*a + y*b);

 if (N < x*a){
    row += N/a;
    row += 1;
    seat = N % a + 1;
 }
 else {
    row += x;
    N -= x*a;
    row += N/b;
    row += 1;
    seat = N % b + 1;
 }
cout << row << " " << seat <<endl;
```

```
 return 0;
}
```

9- 11 класс

A. День рождения

Обозначим число первого вида конфет за x, а второго – за y. Можно заметить, что сумма $0 < (x + y) < 2N$ и делится на N, значит, она равна N. Это значит, что для каждого конкретного значения x мы можем точно сказать, что $y = N - x$. Тогда можно перебрать все x от 0 до d и проверить, что пара $(x, N - x)$ является корректной.
Должны выполняться неравенства:

$$0 \le x \le d,$$
$$0 \le N - x \le t.$$

Тогда

$$0 \le x \le d,$$
$$x \ge N - t.$$

Тогда ответ на задачу – формула $\max(0, d - (N - t) + 1)$.
Пример решения.

```
n = int(input())
d = int(input())
t = int(input())
print(max(0, d - (n - t) + 1))
```

B. Кинотеатр

Эффективное полное решение включает в себя формулы для вычисления номера ряда и номера места. Для удобства номер билета будем отсчитывать с нуля, для этого вычтем из N единицу. Известно, что каждые x+y рядов содержат x*a+y*b мест. Посмотрим, сколько раз укладывается x + y в N. Разделим номер билета N на (x*a + y*b) с остатком. Обозначим частное за d, остаток за r. Рассмотрим остаток r, если остаток r от 0 до x*a – 1, то к номеру ряда добавится add = r / a и ряд будет равен N / (x*a + y*b) * (x+y) + r/a + 1, а место будет равно r % a + 1, иначе добавится add = x + (r – x*a)/b и ряд будет равен N / (x*a + y*b) * (x+y) + add + 1, а место будет равно (r – x*a) % b + 1.

```
#include <iostream>
using namespace std;
int main(){
 long long x, a, y, b, N;
 cin >> x >> a >> y >> b >> N;
 long long row = 0;
 long long seat = 0;

 N -= 1;
 row = N / (x*a + y*b) * (x+y);
 N = N % (x*a + y*b);

 if (N < x*a){
    row += N/a;
    row += 1;
    seat = N % a + 1;
 }
 else {
    row += x;
    N -= x*a;
    row += N/b;
    row += 1;
    seat = N % b + 1;
 }
cout << row << " " << seat <<endl;

 return 0;
}
```

C. Математическая школа

Найдем произведение цифр в числе N. Пусть оно равно n. Нам «выгодно» выделять как можно большие цифры в ответе. Например, если n = 18, то из двух вариантов ответа 29 или 36 меньшим будет ответ 29. При этом в ответе 29 есть цифра 9, и если попробовать выделить сначала большую цифру 9, то после деления 18/9 получится меньший результат, который мы сможем использовать для старших разрядов записи ответа. Получаем «жадное» решение — давайте делить наше число на все возможные цифры от 9 до 2 в порядке уменьшения, то есть попробуем в произведении выделить как можно больше девяток, затем как можно больше восьмёрок и т.д. В конце нужно вывести найденные цифры в обратном порядке. Отдельно нужно обработать случай, когда n = 1, тогда ответ также равен 1.

Пример решения.

```
#include <iostream>
using namespace std;
int k[10];
int main(){
long long N;
 cin >> N;
long long n = 1;
 while (N > 0){
    n *= N % 10;
    N /=10;
 }
if (n == 1) {
 cout << 1;
 return 0;
}
 int d = 9;
 while (d > 1){
   if (n % d == 0) {
        n = n/d; k[d]++;
    }
   else d--;
 }
  for (int i = 2; i <= 9; i++){
    for (int c = 1; c <= k[i]; c++)
        cout << i;
  }
return 0;
}
```

Неэффективное частичное решение можно получить, если перебирать все числа, начиная с 1, пока не найдётся число, произведение цифр которого равно произведению цифр в N.

D. Парк культуры и отдыха

Необходимо быстро вычислять количество символов ‘<’ слева и ‘>’ справа от каждого символа в строке, а затем взять из них минимум, поэтому эффективное решение на полный балл включает себя вычисление префиксных и суффиксных сумм.

```
#include <iostream>
#include <string>
using namespace std;
 int sum_l[100001], sum_r[100001];
int main(){
 int n;
 cin >> n;
 string s;
 cin >> s;

 for (size_t i = 0; i < s.length(); i++){
    sum_l[i] = (i > 0)?sum_l[i - 1]: 0;
    if (s[i] =='<')
     sum_l[i] ++;

 }

 for (int i = s.length() - 1; i >= 0; i-- ){
     sum_r[i] = (i <  s.length() - 1)?sum_r[i+1]:0;

     if (s[i] =='>')
      sum_r[i] ++;

 }

 int ans = 100001;
 for (size_t i = 0; i < s.length(); i++){
    if (i >= 1 && i < s.length() - 1)
     ans = min(ans, sum_l[i-1] + sum_r[i + 1]);
    else
        if (i>=1)
             ans = min(ans,  sum_l[i - 1]);
    else
            ans = min(ans,   sum_r[i + 1]);

 }
cout << ans << endl;
return 0;

}
```

E. Кондитерские

Эффективное решение на 100 баллов включает в себя бинарный поиск по ответу. Заметим, что при увеличении расстояния, количество тортов, которые можно доставить из кондитерских на расстоянии не больше данного, строго возрастает. Поэтому можно для каждой кондитерской запустить бинарный поиск по ответу. Для того, чтобы быстро пересчитать сумму на отрезке [l; r], можно использовать префикс-суммы. Пусть sum_i — сумма s_j с 1-го по i-й. Для удобства положим $sum_0 = 0$. Тогда если у i-й кондитерской радиус доставки равен x, то на ее витрину войдет $sum_{min(i+x+1,n)} - sum_{max(i-x,0)}$ тортов.

```
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

int main() {
    int n;
    long long R;

    cin >> n >> R;

    vector <long long> a(n + 1), pr(n + 1, 0);
    for (int i = 1; i <= n; i++) {
        cin >> a[i];
        pr[i] = pr[i - 1] + a[i];
    }

    for (int i = 1; i <= n; i++) {
        if (a[i] > R) {
            cout << "-1\n";
            continue;
        }

        int l = 0, r = max(i - 1, n - i) + 1;
        while (r - l > 1) {
            int m = (l + r) / 2;
            int sl = max(1, i - m), sr = min(n, i + m);
            if (pr[sr] - pr[sl - 1] <= R)
                l = m;
            else
                r = m;
        }

        cout << l << '\n';
    }

    return 0;
}
```